# Homework 5 - Método Monte-Carlo

Claudia Lizeth Hernández Ramírez

29 de septiembre de 2021

**Resumen**

Se identifcó que conforme aumenta el grupo, es decir, la cantidad de puntos, la probabilidad de obtener dígitos iguales a los del resultado obtenido en `Wolfram Alpha` es mayor.

## 1. Introducción

En esta tarea se estudiará la precisión del estimado del integral con en método Monte Carlo, comparado con el valor producido por Wolfram Alpha, en términos del número de decimales correctos, aumentando el tamaño de muestra.

## 2. Desarrollo

Como se explicó en la introducción, el propósito de esta tarea es comparar la cantidad de decimales que resultan igual en el valor obtenido de la página `Wolfram Alpha` y el que obtenemos del código base, que vimos en clase[2].

Comencé generando un pequeño programa el cuál compararía la cantidad de decimales entre un número y otro utilizando `strings`.

Listing 1: Código para comparar números como strings.

```
suppressMessages (library(tidyverse))
numewolf = 0.369258147 #numero obtenido de WolframAlpha
numerock  = 0.365257741 #numero obtenido en el programa
as.character(numewolf)
as.character(numerock)
datos = data.frame()

posicion = 3
str_sub(as.character(numewolf), 3, posicion)
str_sub(as.character(numerock), 3, posicion)

while (str_sub(as.character(numewolf), 3, posicion) ==
       str_sub(as.character(numerock), 3, posicion)) {
  print(posicion-2)
  posicion = posicion + 1
}
datos = (rbind(datos, posicion-3))
```

Posteriormente, era necesario incluir ese código en el que vimos en clase [2], por lo que era necesario hacer algunos cambios para que pudiera ser compatible; Por mencionar algunos, `numerock` tomaría el valor del resultado de la integral que arroja el código base y `numewolf = 0.048834`. Estos resultados fueron guardados en una data frame llamado `compara`

Listing 2: Código data frame `compara`.

```
while (str_sub(as.character(numewolf), 3, posicion) ==
         str_sub(as.character(numerock), 3, posicion)) {
    print(posicion-2)
    posicion = posicion + 1
  }
  datos = c(i, posicion-3)
  compara = (rbind(compara, datos))
  names(compara) = c("Cantidad", "DecimalesCorrectos")
}
```

Antes de hacer pruebas estadísticas es necesario realizar pruebas de normalidad a nuestros datos para decidir la prueba que utilizaremos, en este caso como las muestras no exceden de las 50 repeticiones, utilizaremos la prueba de normalidad de **Shapiro Wilk**.

Cuadro 1: Resultados obtenidos de prueba de normalidad de Shapiro.

| W | P |
|---|---|
| 0.9035 | $1{,}395 \times 10^{-11}$ |

Listing 3: Código prueba de normalidad Shapiro-Wilk.

```
#Estadistica Shapiro Wilk
compara
str(compara)
names(compara)
shapiro.test(compara$DecimalesCorrectos)
```

Planteando que:

- **Hipótesis nula**: la distribución de la variable es normal.

- **Hipótesis alternativa**: la distribución de la variable no es normal.

En la literatura podemos encontrar que diversos autores establecen que en pruebas de normalidad para aceptar la hipótesis nula el valor de `P` debe ser mayor al 0.05, es decir mayor al 5 %. De los resultados mostrados en el cuadro 1, observamos que nuestros valores de `P` son por mucho, menores a 0.05, por lo tanto tenemos pruebas suficientes para **rechazar** nuestra hipótesis nula.

Ahora, sabiendo que la distribución de nuestros datos no son **normales**, que tenemos más de dos grupos por analizar y además son muestras independientes, trabajaremos con la prueba de **Kruskal-Wallis**[1].

Cuadro 2: Resultados obtenidos de prueba Kruskal-Wallis.

| Chi cuadrada | DF | P |
|---|---|---|
| 149.23 | 4 | $2{,}2 \times 10^{-16}$ |

Listing 4: Código prueba estadística Kruskal-Wallis.

```
#Estadistica Kruskal Wallis
compara %>%
  group_by(Cantidad) %>%
  summarise(
    promedio = mean(DecimalesCorrectos, na.rm = TRUE),
    desviacion_std = sd(DecimalesCorrectos, na.rm = TRUE),
    varianza = sd(DecimalesCorrectos, na.rm = TRUE)^2,
    mediana = median(DecimalesCorrectos, na.rm = TRUE),
    rango_intercuartil = IQR(DecimalesCorrectos, na.rm = TRUE)
  )

kruskal.test(DecimalesCorrectos ~ Cantidad, data = compara)
```

```
pairwise.wilcox.test(compara$DecimalesCorrectos, compara$Cantidad) #diferencias
    entre grupos
```

Cuadro 3: Diferencias entre grupos. Kruskal-Wallis.

| | 5k | 50k | 500k | 5M |
|---|---|---|---|---|
| 50k | 0.00037 | | | |
| 500k | $1{,}1 \times 10^{-10}$ | 0.00046 | | |
| 5M | $3{,}2 \times 10^{-15}$ | $2{,}0 \times 10^{-9}$ | 0.00076 | |
| 50M | $2{,}0 \times 10^{-16}$ | $5{,}0 \times 10^{-15}$ | $4{,}6 \times 10^{-10}$ | 0.00038 |

Del cuadro 3 podemos deducir que existen diferencias estadísticamente significativas entre los grupos, lo cual es comprobado visualmente en la figura 1.

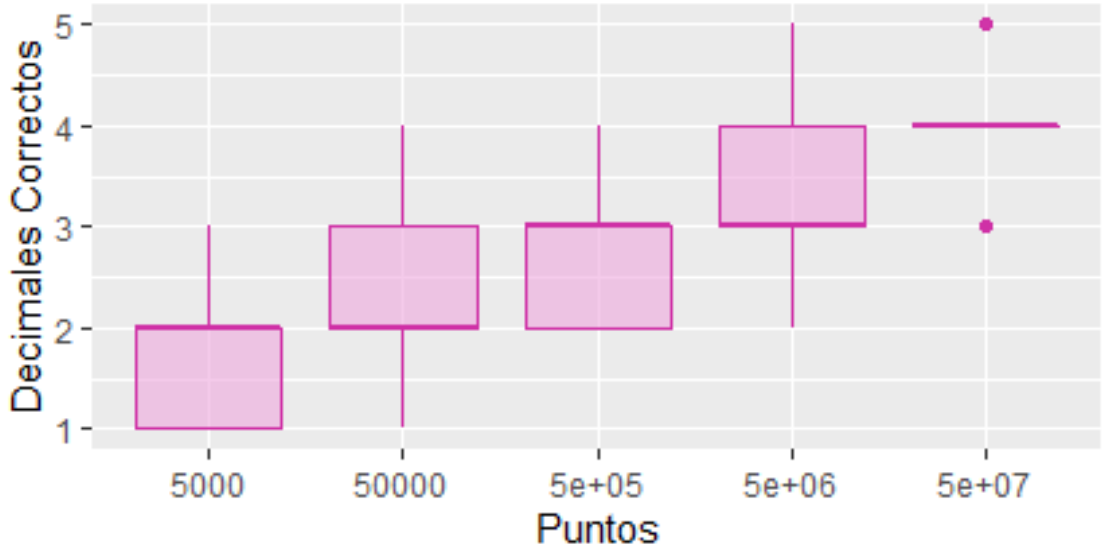


Figura 1: Variación de la cantidad de decimales correctas respecto al número de puntos.

Listing 5: Código data frame `compara`.

```
# Boxplot
compara$Cantidad = as.factor(compara$Cantidad)
ggplot(compara, aes(x=Cantidad, y=DecimalesCorrectos)) +
  geom_boxplot(fill = "#efa9dd", color = "#cf30a6", alpha = 0.6) +
  labs(x="Puntos", y="Decimales Correctos")
```

## 3. Conclusión

Teniendo que:

-Hipótesis nula: las medias de población son todas iguales.

-Hipótesis alternativa: las medias de población no son todas iguales.

Y con un nivel de significancia = 0.05. Podemos rechazar la hipótesis nula, por lo tanto las diferencias entre algunas medianas son estadísticamente significativas. Se identifcó que conforme aumenta el grupo, es decir, la cantidad de puntos, la probabilidad de obtener dígitos iguales a los del resultado obtenido en `Wolfram Alpha` es mayor.

# 4. RETO 1 - PI.

# 5. Desarrollo - Reto 1

Se utlizó como código base el visto en clase[3]. Utilizando la misma lógica que en la tarea base, escribí un código el cuál compararía la cantidad de decimales entre un número y otro `strings`.

Listing 6: Código para comparar números como strings.

```
suppressMessages (library(tidyverse))
resultado = pi
    at = as.character(truepi)
    ar = as.character(resultado)
    posicion = 3

    while (str_sub(at, 3, posicion) ==
           str_sub(ar, 3, posicion)) {
      print(posicion-2)
      posicion = posicion +1
    }
    data = c(muchos, posicion-3)
    datos = rbind(datos, data)
    names(datos) = c("Muestra", "DeCorrect")
```

Nuevamente lo agregué a [3]. Los resultados que generaba el programa los guardé en una data frame llamado `datos`

Realicé pruebas de normalidad a los datos para decidir la prueba estadística, en esta ocación las muestras no exceden de las 20 repeticiones, por lo tanto utilicé la prueba de normalidad de `Shapiro Wilk`.

Cuadro 4: Resultados obtenidos de prueba de normalidad de Shapiro, reto 1.

| W | P |
|---|---|
| 0.68651 | $4{,}92 \times 10^{-10}$ |

Listing 7: Código prueba de normalidad Shapiro-Wilk.

```
#Estadistica Shapiro Wilk
datos
str(datos)
names(datos)
shapiro.test(datos$DeCorrect)
```

Planteando que:

- `Hipótesis nula`: la distribución de la variable es normal.

- `Hipótesis alternativa`: la distribución de la variable no es normal.

Ya que `P` es menor a 0.05, rechazamos hipótesis nula. Trbajaremos con Kruskal Wallis.

Cuadro 5: Resultados obtenidos de prueba Kruskal-Wallis, reto 1.

| Chi cuadrada | DF | P |
|---|---|---|
| 4.3365 | 2 | 0.1144 |

Listing 8: Código prueba estadística Kruskal-Wallis.

```
#Estadistica Kruskal Wallis
datos %%
  group_by(Muestra) %%
```

```
  summarise(
    promedio = mean(DeCorrect, na.rm = TRUE),
    desviacion_std = sd(DeCorrect, na.rm = TRUE),
    varianza = sd(DeCorrect, na.rm = TRUE)^2,
    mediana = median(DeCorrect, na.rm = TRUE),
    rango_intercuartil = IQR(DecCorrect, na.rm = TRUE)
  )

kruskal.test(DeCorrect ~ Muestra, data = datos)
pairwise.wilcox.test(datos$DeCorrect, datos$Muestra) #diferencias entre grupos
```

Cuadro 6: Diferencias entre grupos Kruskal-Wallis , reto 1.

| | 10k | 50k |
|---|---|---|
| 50k | 0.24 | |
| 100k | 0.16 | 0.64 |

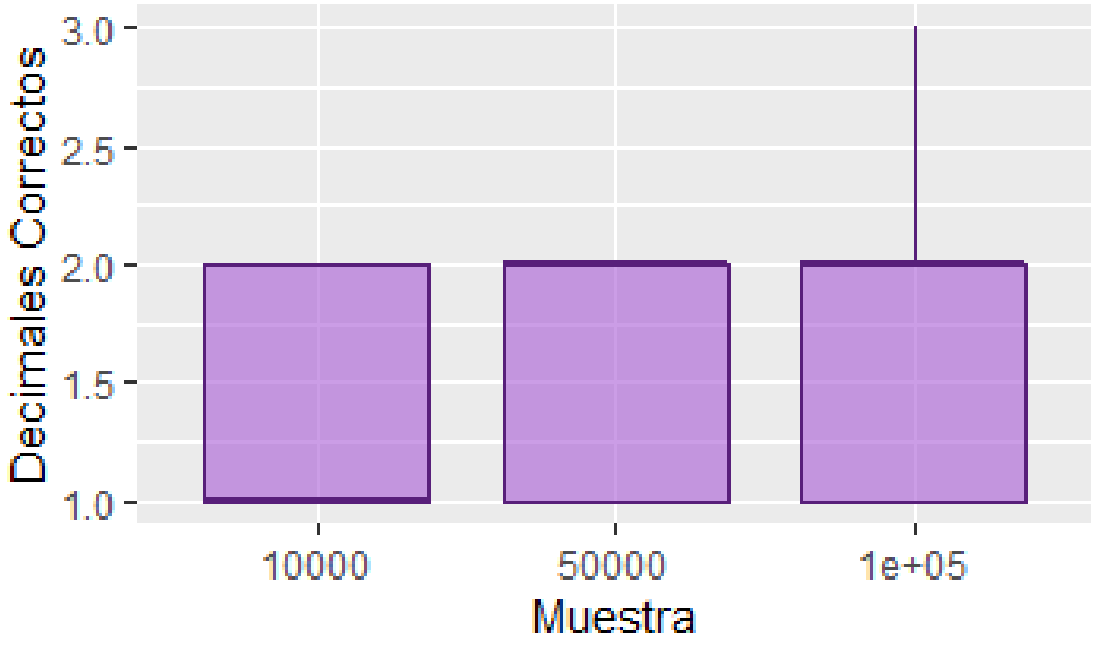


Figura 2: Variación de la cantidad de decimales correctas respecto al número de puntos, reto 1.

Listing 9: Código data frame `datos`.

```
# Boxplot
datos$Muestra = as.factor(datos$Muestra)
ggplot(datos, aes(x=Muestra, y=DeCorrect)) +
  geom_boxplot(fill = "#a85cd6", color = "#581f7a", alpha = 0.6) +
  labs(x="Muestra", y="Decimales Correctos")
```

# 6. Conclusión - Reto 1

Teniendo que:

-Hipótesis nula: las medias de población son todas iguales.

-Hipótesis alternativa: las medias de población no son todas iguales.

Y con un nivel de significancia = 0.05. No podemos rechazar la hipótesis nula, por lo tanto las diferencias entre las medianas no son estadísticamente significativas.

# Referencias


[1] E. Flores Ruiz. Selección de pruebas de estadísticas, 2017. URL `http://www.scielo.org.mx/pdf/ram/v64n3/2448-9190-ram-64-03-0364.pdf`.

[2] Elisa Schaeffer. Codigo clase, 2021. URL `https://github.com/satuelisa/Simulation/blob/master/MonteCarlo/integral.R`.

[3] Elisa Schaeffer. Codigo clase reto, 2021. URL `https://github.com/satuelisa/Simulation/blob/master/MonteCarlo/pi.R`.